Amendments to the Specification:

Please replace the paragraph beginning at page 2, line 13 with the following:

Referring to FIG. 1, a communication system 10 includes a parallel, hardware-based multithreaded processor 12. The hardware-based multithreaded processor 12 is coupled to a bus such as a PCI bus 14, a memory system 16, and a second bus 18. The processor 12 includes a bus interface 28 that couples the processor 12 to the second bus 18. Bus interface 28 in one embodiment couples the processor 12 to the so-called FBUS 18 (FIFO (first-in, first-out) bus). The FBUS interface (FBI) 28 is responsible for controlling and interfacing the processor 12 to the FBUS 18. The FBUS 18 is a 64-bit wide FIFO bus, used to interface to MAC devices 13. The system 10 is especially useful for tasks that can be broken into parallel subtasks or functions. Specifically, a hardware-based multithreaded processor 12 is useful for tasks that are bandwidth oriented rather than latency oriented. The hardware-based multithreaded processor 12 has multiple microengines 22 each with multiple hardware controlled threads that can be simultaneously active and independently work on a task.

Please replace the paragraph beginning at page 3, line 10 with the following:

The hardware-based multithreaded processor 12 also includes a central controller (also called processor or microprocessor) 20 that assists in loading microcode control for other resources of the hardware-based multithreaded processor 12 and performs

other general purpose computer type functions such as handling protocols, exceptions, and extra support for packet processing where the microengines 22 pass the packets off for more detailed processing such as in boundary conditions. In one embodiment, the processor 20 is a Strong Arm® (Arm is a trademark of ARM Limited, United Kingdom) based architecture. The general purpose microprocessor 20 has an operating system. Through the operating system, the processor 20 can call functions to operate on microengines 22a-22f. The processor 20 can use any supported operating system, preferably a real time operating system. For the core processor 20 implemented as a Strong Arm architecture, operating systems such as, MicrosoftNT real-time, VXWorks and µCUS, a freeware operating system available over the Internet, can be used.

Please replace the paragraph beginning at page 6, line 17 with the following:

Each of the functional units, e.g., the FBI 28, the SRAM controller 26b, and the SDRAM controller 26a, are coupled to one or more internal buses. The internal buses are dual, 32-bit buses (i.e., one bus for read and one for write). The hardware-based multithreaded processor 12 also is constructed such that the sum of the bandwidths of the internal buses in the processor 12 exceeds the bandwidth of external buses coupled to the processor 12. The processor 12 includes an internal core processor bus 32, e.g., an ASB bus (Advanced System Bus), that couples the processor core 20 to the memory controllers 26a, 26b and to an ASB translator 30. The ASB bus 32 is a subset of the so-called AMBA bus that is used with the Strong Arm processor core. The processor 12 also includes a private bus 34 that

couples the microengine units 22 to SRAM controller 26b, ASB translator 30, and FBI 28. A memory bus 38 couples the memory controllers 26a, 26b to the bus interfaces 24 and 28 and memory system 16 including a flashrom 16c used for boot operations and so forth.

Please replace the paragraph beginning at page 11, line 18 with the following:

The FBI 28 checks 66 the requested bytes by looking for an end-of-packet indicator at the end of the first 64 bytes. If the packet is a minimum length packet 68, the FBI 28 begins 70 its next operation, having received 64 a complete packet of data and its status. If the packet is between 64 and 72 bytes, the FBI 28 requests 72 another eight bytes so as to receive 72 ~~52~~ the packet status four cycles later. These eight additional bytes are stored in the second status quadword for that element in the receive FIFO 29a. The FBI 28 begins 70 its next operation having now received a complete packet of data and its status.

Please replace the paragraph beginning at page 12, line 4 with the following:

Still referring to FIG. 4B, the FBI 28 may operate in Fetch_10 mode. Fetch_10 mode optimizes bandwidth for a high frequency of packets having between 64 and 80 ~~72~~ bytes, e.g., packets with VLAN (virtual local area network) tags from the Gigabit Ethernet device 13b. In Fetch_10 mode, the FBI 28 requests 62 and receives 64 bytes as described above in FIG. 4B, except the FBI 28 issues 62 ten requests, each for M bytes, over

ten clock cycles (one request per cycle). The first 64 bytes are stored in one element of the receive FIFO 29a, bytes 65-72 in the first status quadword of that element, and bytes 73-80 in the second status quadword of that element. As above, the FBI 28 checks 66, 68 to see if the receive FIFO 29a contains the packet data and its status. If so, the FBI 28 begins 48 its next operation 70. If not, it requests 72 another M bytes. Once received, this quadword may be stored in a third status quadword of that element or in another receive FIFO 29a element. The FBI 28 then begins 70 its next operation having now received a complete packet of data and its status.